Chapter 10

Facial Animation

Chapter 10

Facial models

Anatomic structure
FACS
Facial models
- rigid parts
- texture maps
- continuous surface models
- textures

Facial Animation by Texture Maps

Ebert - from *Getting Into Art*

Facial Animation

http://mambo.ucsc.edu/psl/sig97/siggraph97-panel.html

Facial Animation

Complex surface

Very familiar structure

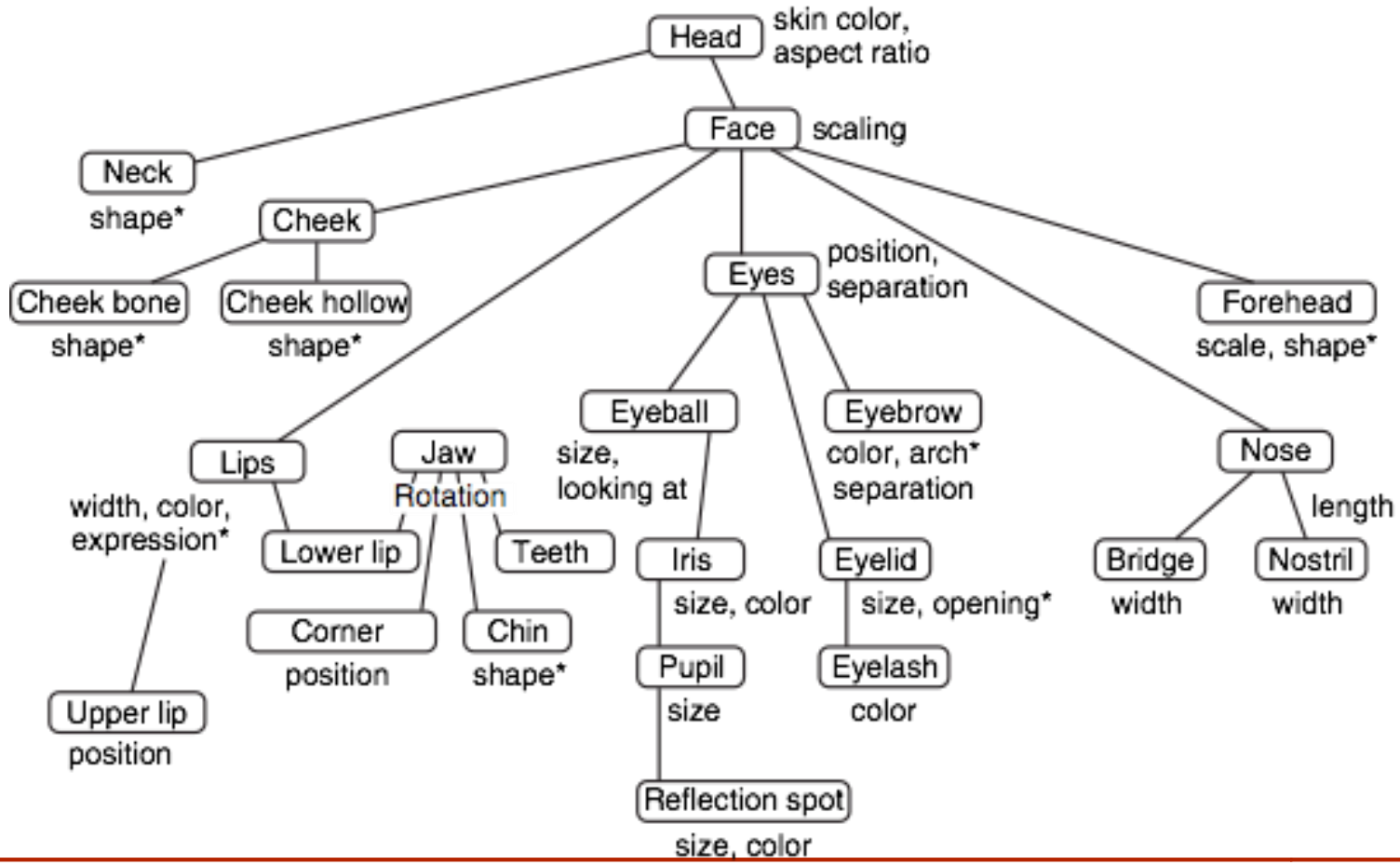
Deformable surface

Very important - Principal means of communication

Facial expressions, lip-synch, prosodic facial animation

Chapter 10

Facial models



Facial Action Coding System (FACS)

Factial Action Coding System by Eckman and Friesen breaks facial movements down into basic **action units (AU).**

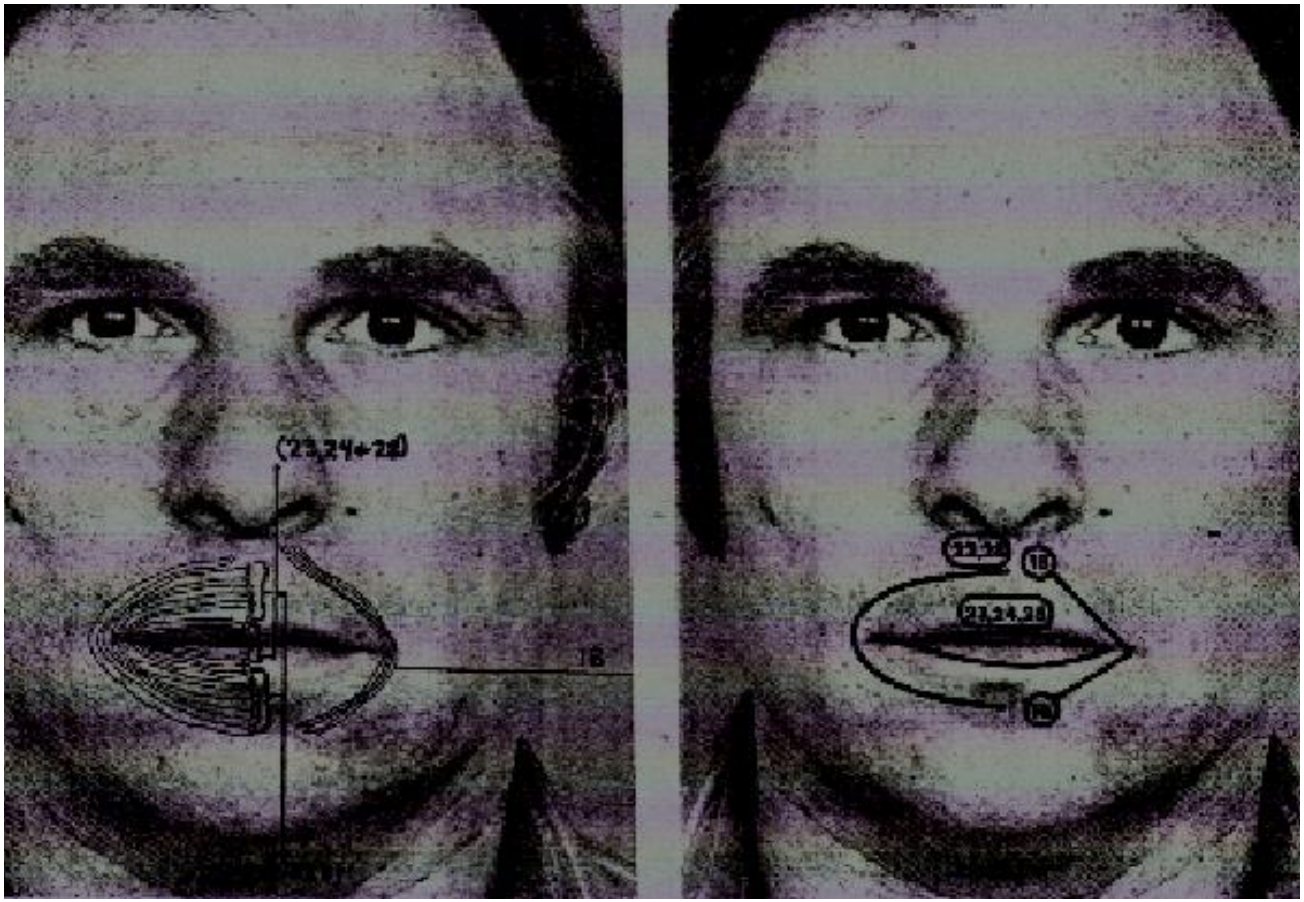
Forty-six Aus are defined in the study by Eckman and Friesen, such as lowering brow, raising inner brow, wink, raising cheek, raise upper lip, or drop jaw.

Parametric value controls enables the system to combine different AUs to form a facial expression by using interpolation.

This system was designed for static expressions and not necessarily animation.

Forming phonemes was not part of this system.

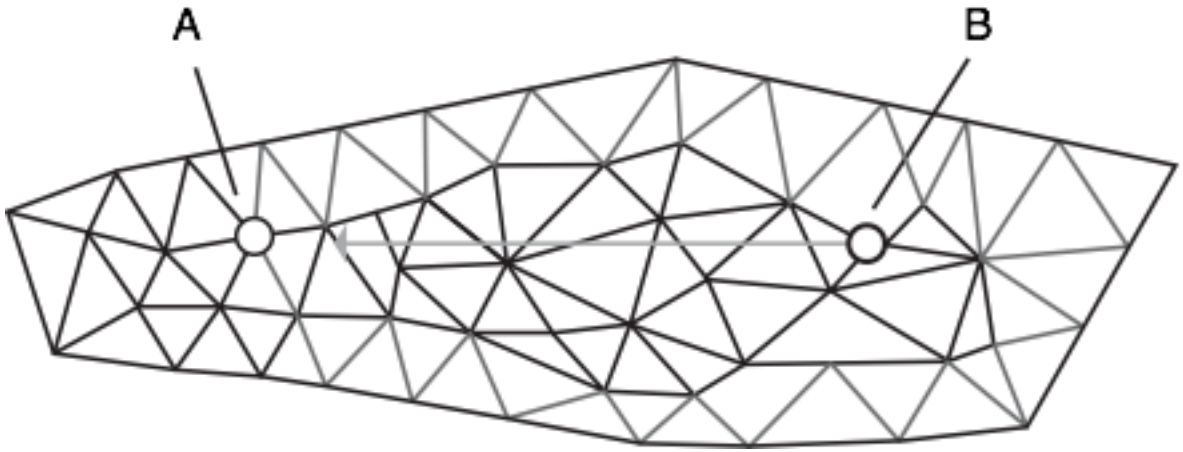
Facial Action Coding System (FACS)



From Eckman and Friesen, 1978

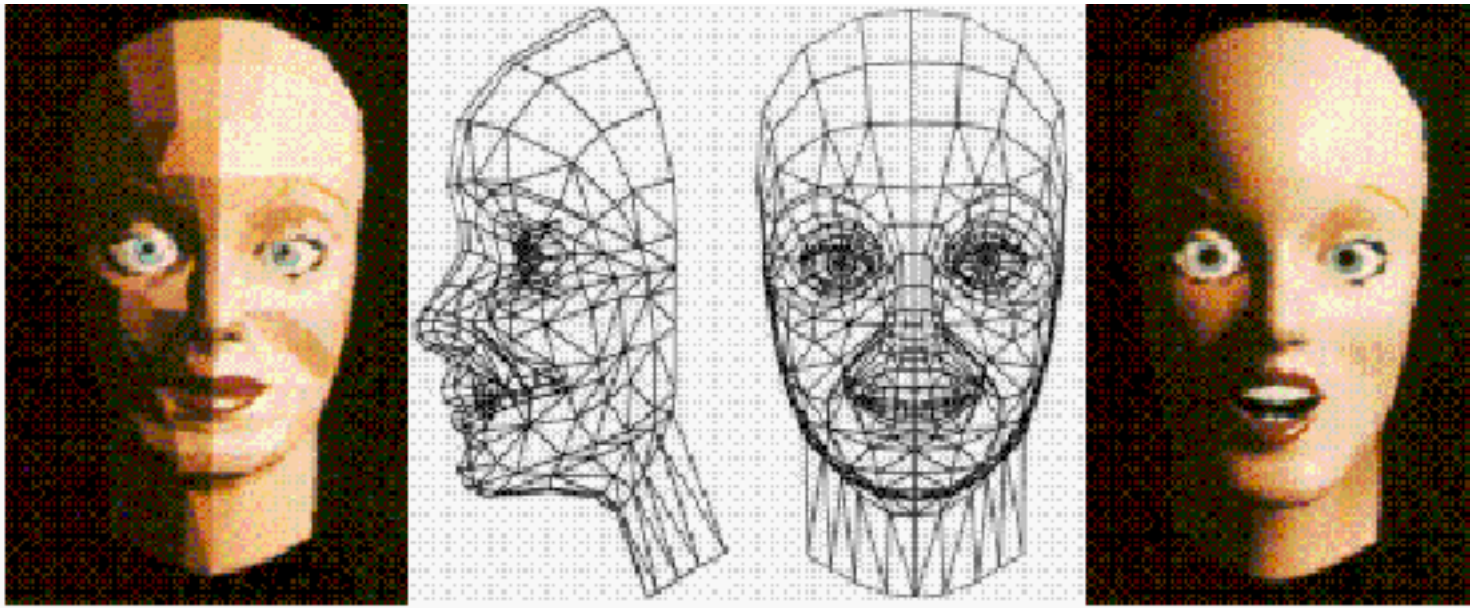
Chapter 10

Facial models



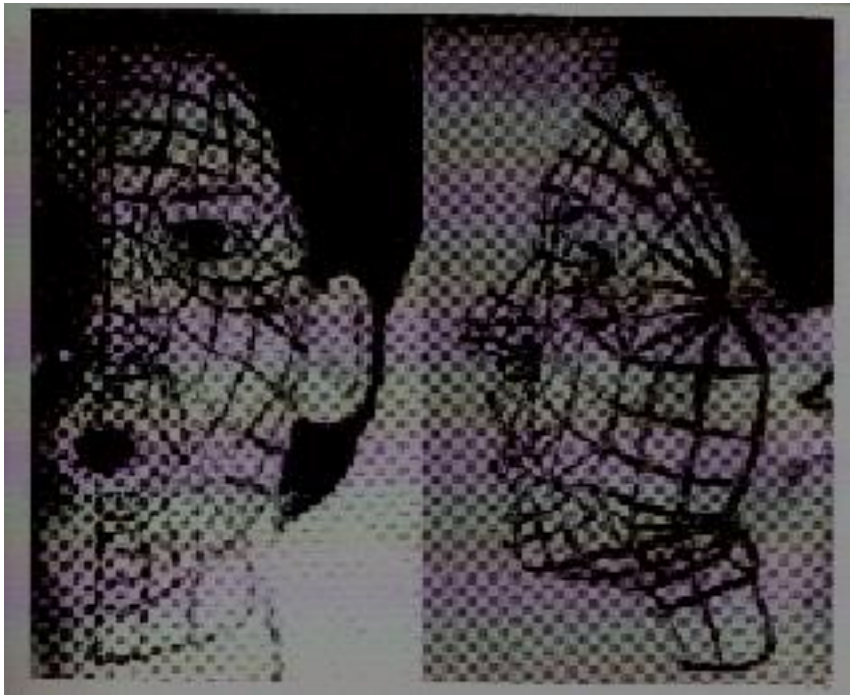
Facial Animation

Fred Parke model



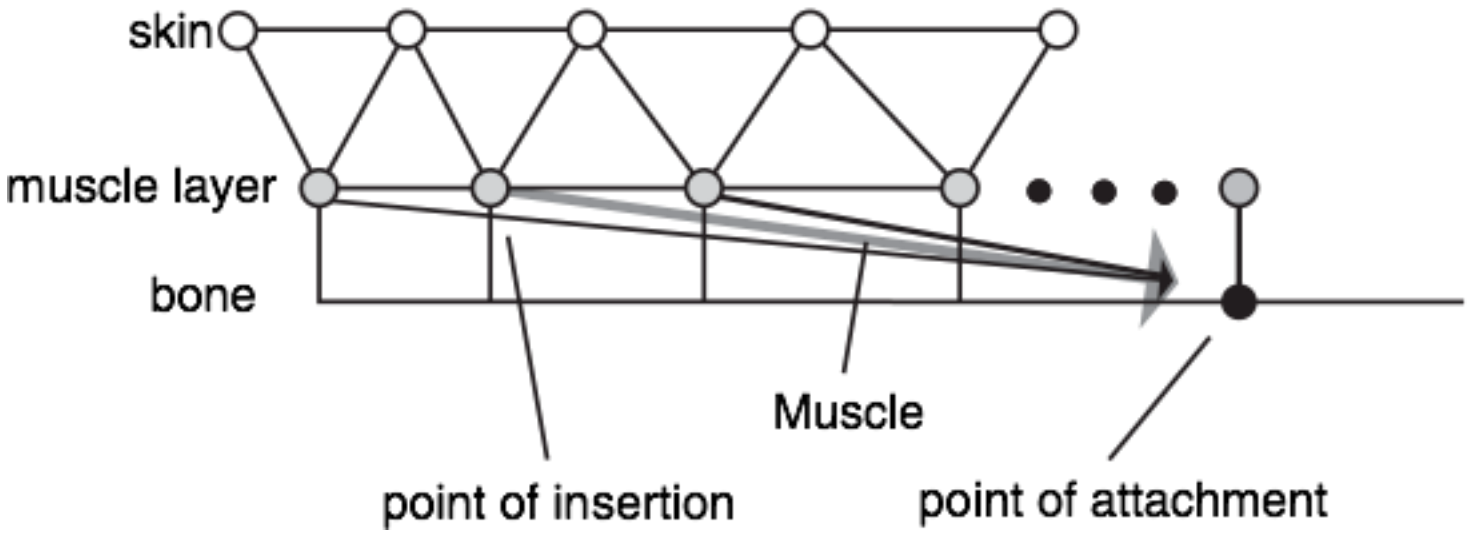
http://mambo.ucsc.edu/psl/sig97/siggraph97-panel.html

Digitizing Faces



Chapter 10

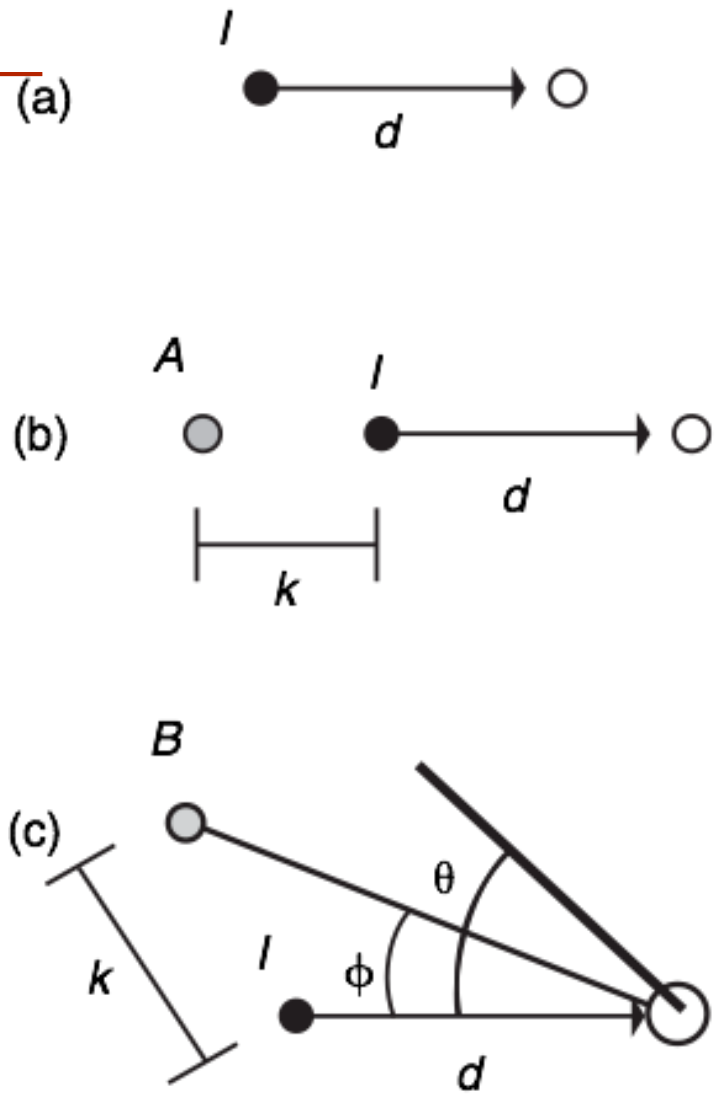
Facial models



Chapter 10

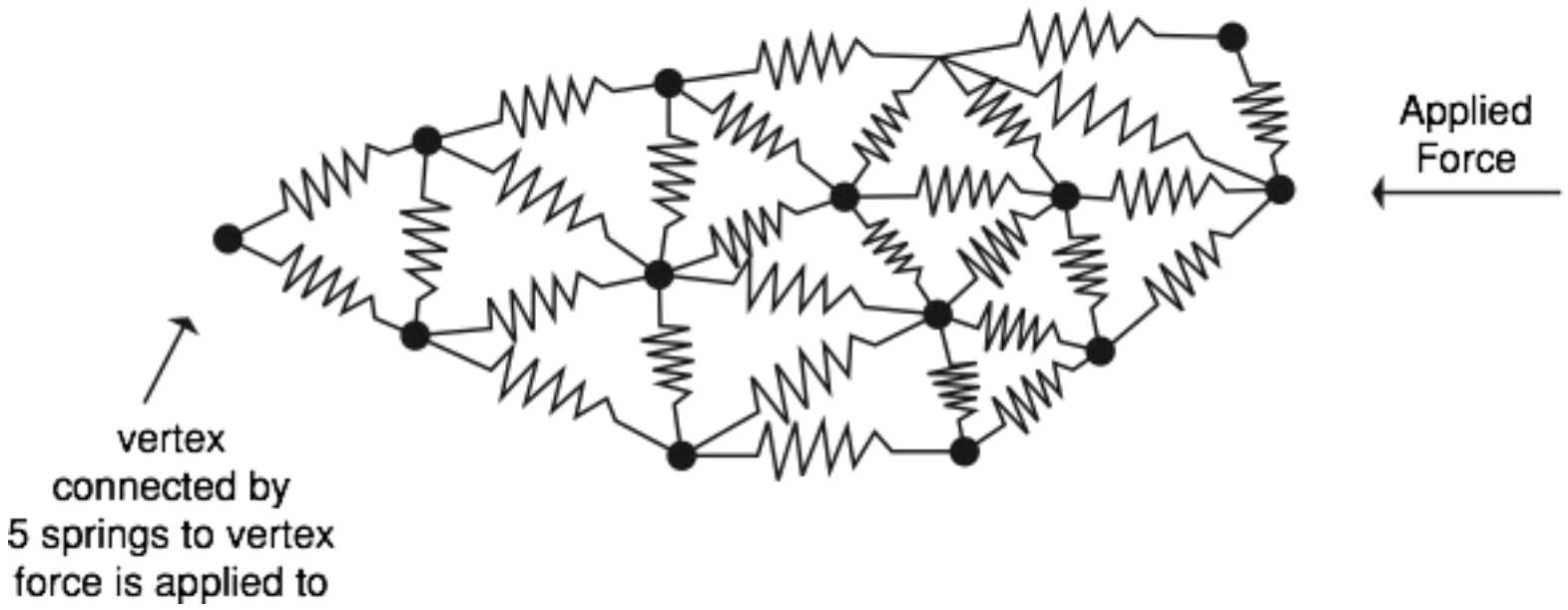
Facial models

Propagation of deformation of the skin can be based on a simple distance-based model such that there is increasing attenuation with increasing distance. This can result in direct movement (a), attenuation based on linear distance (b), or distance and angular deviation (c).



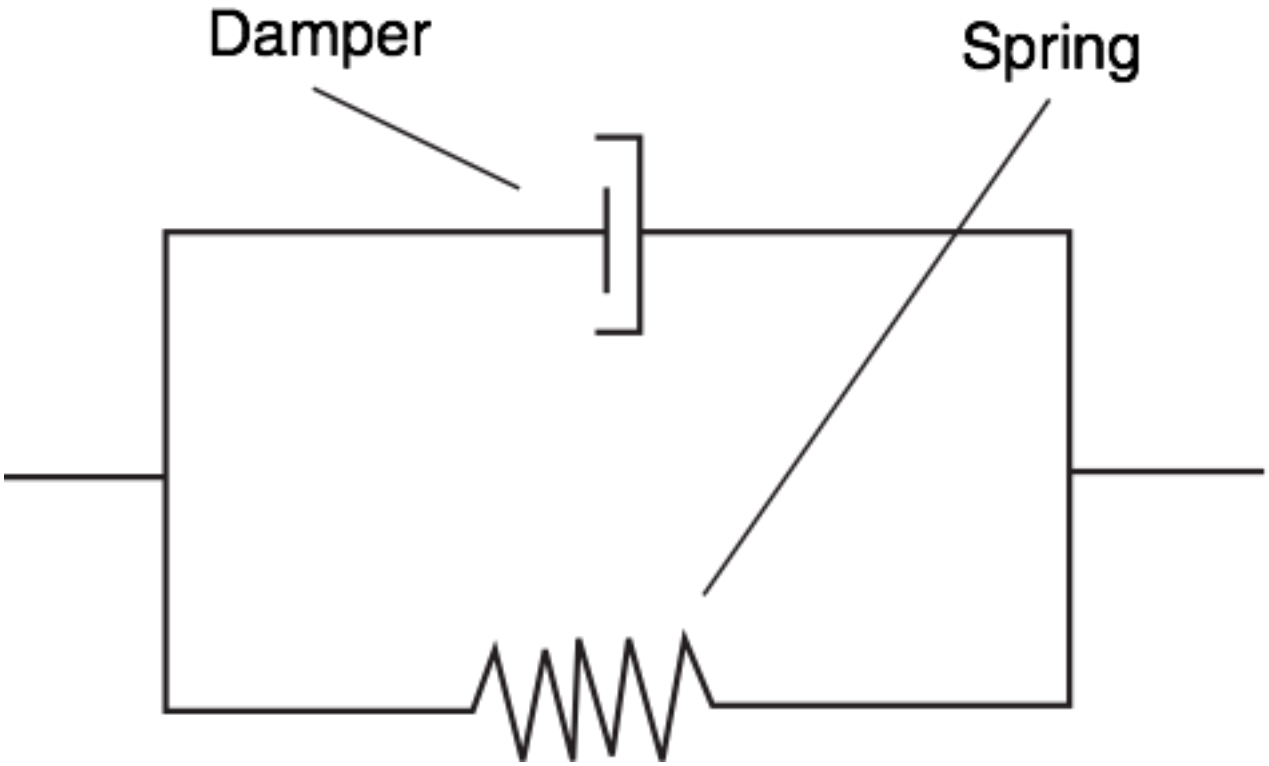
Chapter 10

Facial models



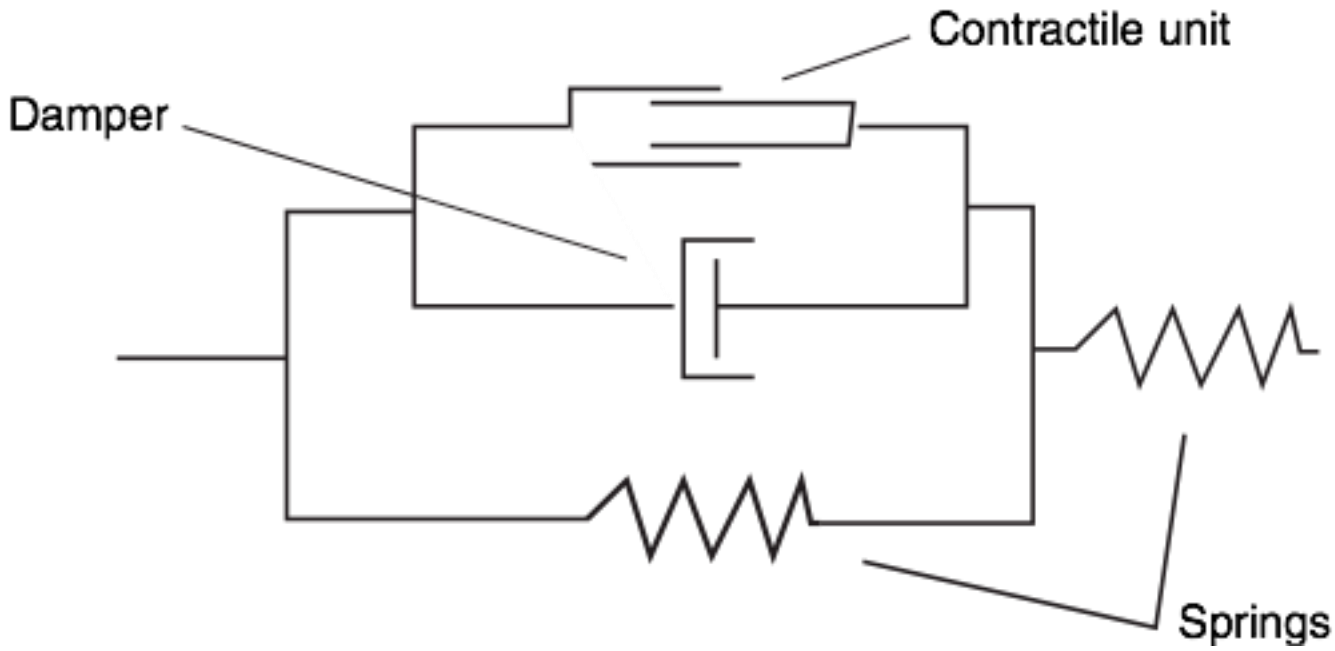
Chapter 10

Facial models



Chapter 10

Facial models

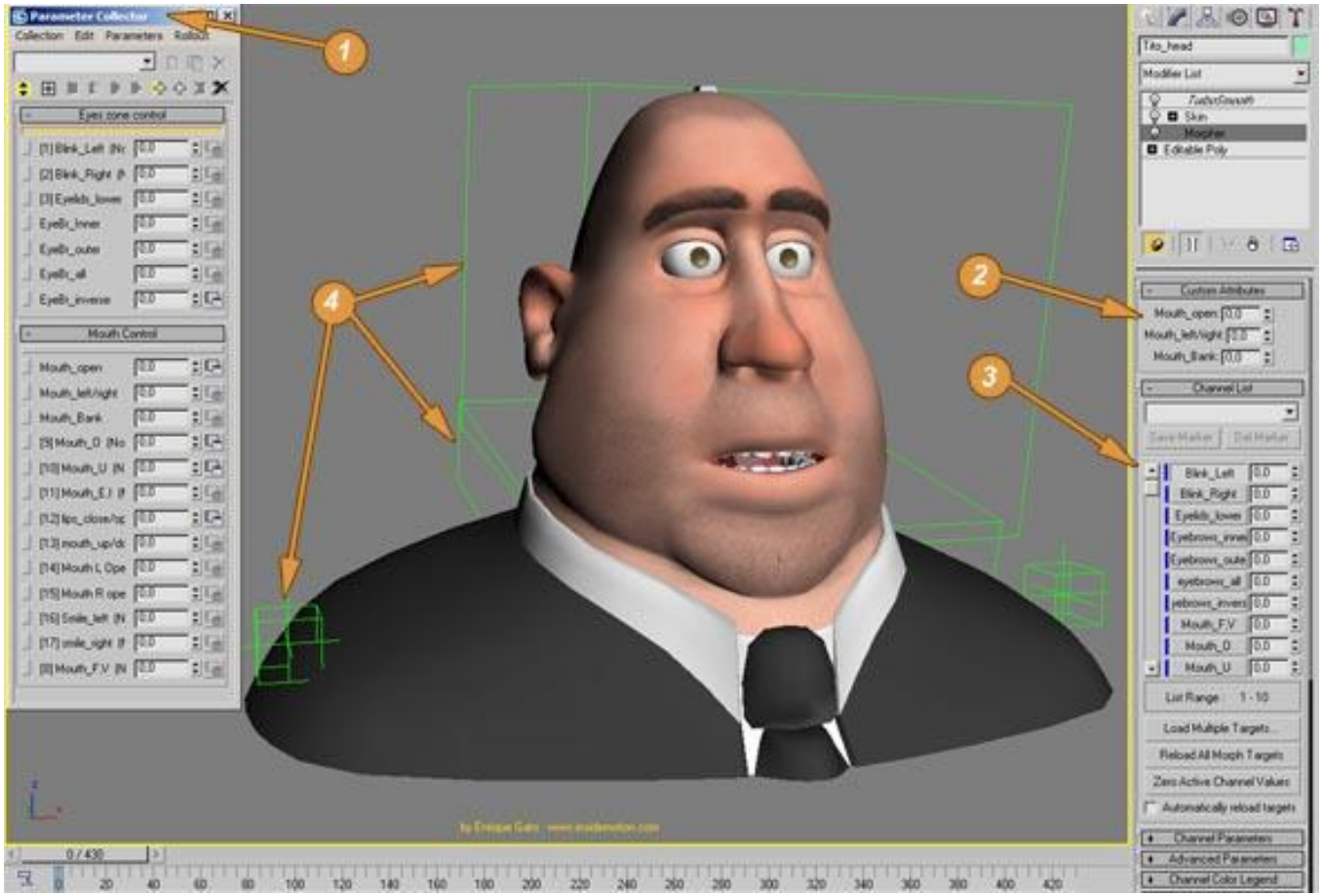


Chapter 10

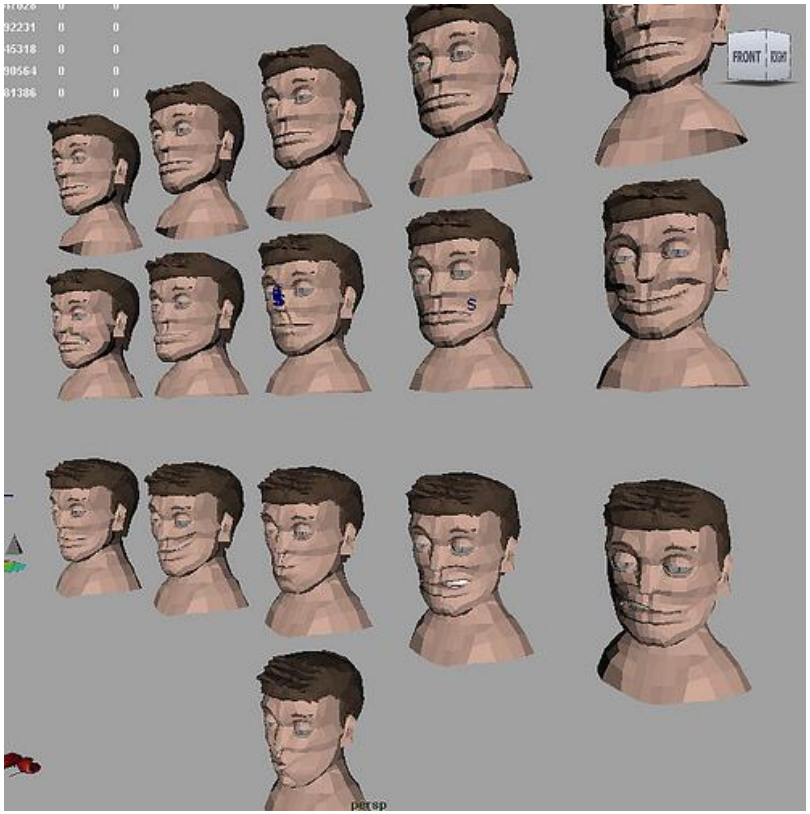
Animating the face

Parameterized models
Blend shapes
Muscle models
Expressions

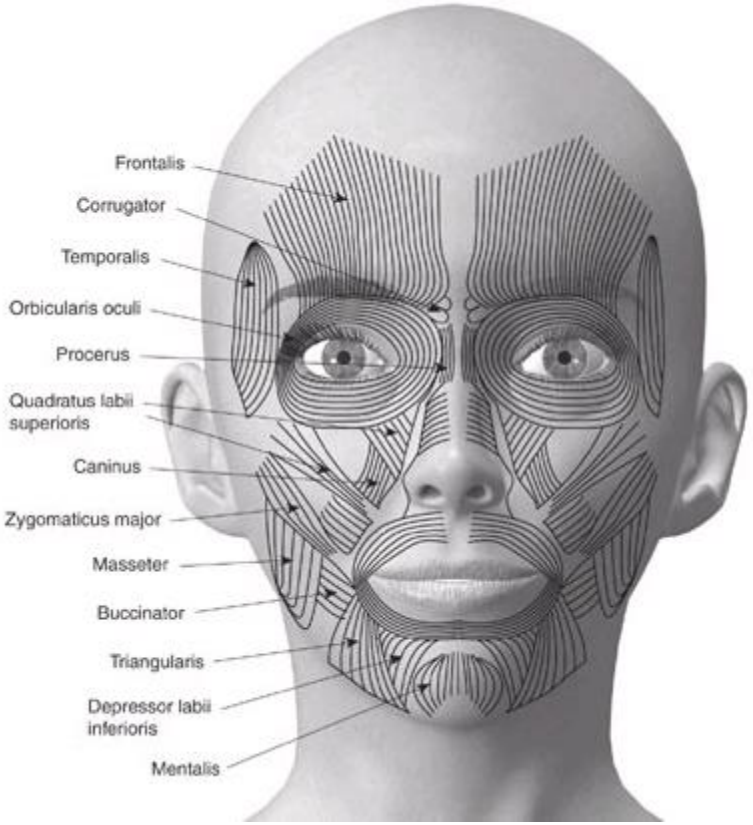
Parameterized Facial features



Facial blend shapes



Facial muscle model



Surface muscle model

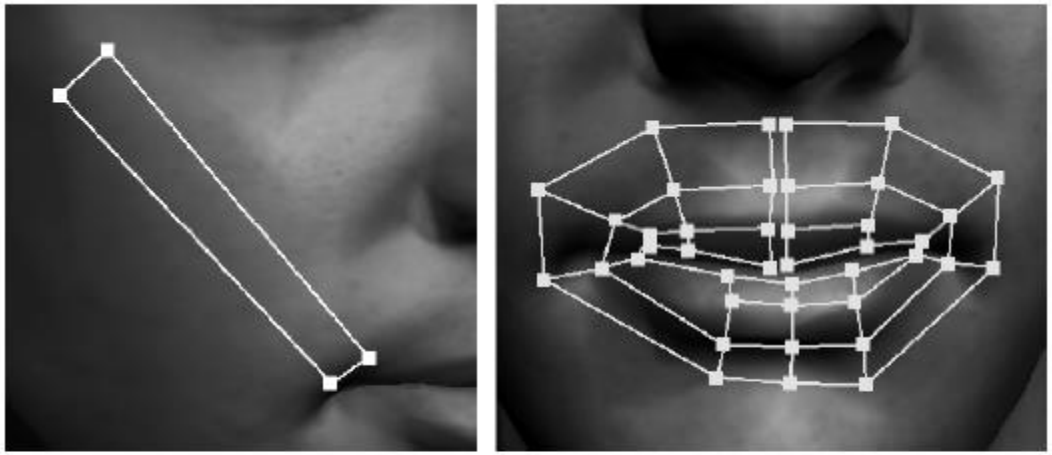


Figure 8: A simple grid (left, zygomatic major*) and a non-uniform complex grid (right,* orbicularis oris*).*

Geometry-based Muscle Modeling for Facial Animation
Kolja K¨ahler J¨org Haber Hans-Peter Seidel

"computer animation" facial

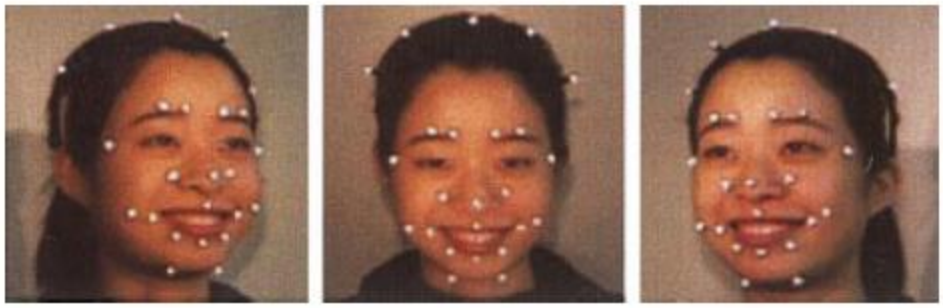


Figure 6. Snapshots taken simultaneously from three video cameras.

Facial Animation

Terzopolous, NYU

Starting with a structured facial mesh, an algorithms that automatically construct functional models of the heads of human subjects from laser-scanned range and reflectance data. These algorithms automatically insert contractile muscles at anatomically correct positions within a dynamic skin model and root them in an estimated skull structure with a hinged jaw. They also synthesize functional eyes, eyelids, teeth, and a neck and fit them to the final model. The constructed face may be animated via muscle actuations.

http://mambo.ucsc.edu/psl/sig97/siggraph97-panel.html

Chapter 10

Lip Sync

Articulators of speech
Phonemes
Coarticulation
Prosody

"computer animation" facial

Text to speech

Audio to speech

} Break down to phonemes

Phonemes to mouth shapes (visemes)

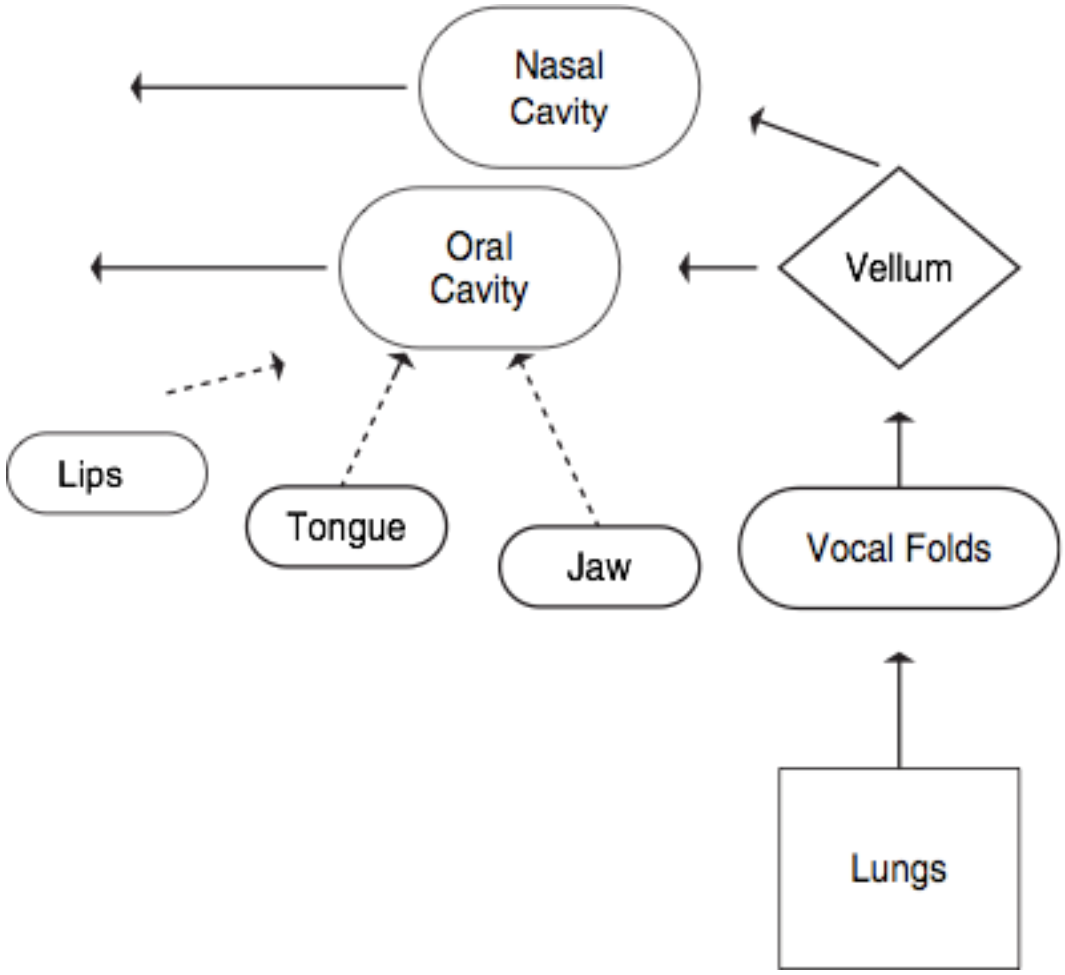
coarticulation

prosody

http://www.youtube.com/watch?v=fxADT-kZNrA

Chapter 10

Lip Sync



Facial Animation

https://accad.osu.edu/research/projectgallery/expressive-facial-animation-model

Facial Animation

Audio Analysis

- Want to capture visual dynamics of speech
- Phonemes are not enough
- Consider *coarticulation*
- Lip shapes for many phonemes are modified based on phoneme's context (e.g. /T/ in "beet" vs. /T/ in "boot")

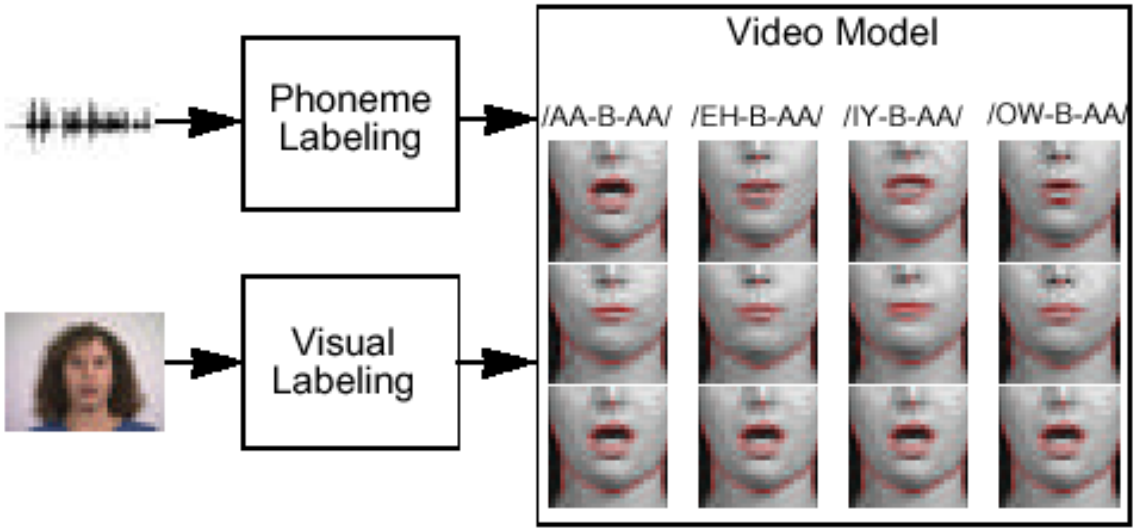
Remaining slides courtesy of Shahzad Malik

Audio Analysis (continued)

- Segment speech into triphones
- e.g. “teapot” becomes /SIL-T-IY/, /T-IY-P/, /IY-P-AA/, /P-AA-T/, and /AA-T-SIL/)
- Emphasize middle of each triphone
- Effectively captures forward and backward coarticulation

Audio Analysis (continued)

- Training footage audio is labeled with phonemes and associated timing
- Use gender-specific HMMs (Hidden Markov model) for segmentation
- Convert transcript into triphones



Synthesis Stage

- Given some new speech utterance
  - Mark it with phoneme labels
  - Determine triphones
  - Find a video example with the *desired transition* in database
- Compute a matching distance to each triphone:

Viseme Classes

- Cluster phonemes into *viseme* classes
- Use 26 viseme classes (10 consonant, 15 vowel):
- (1) /CH/, /JH/, /SH/, /ZH/
- (2) /K/, /G/, /N/, /L/
- …
- (25) /IH/, /AE/, /AH/
- (26) /SIL/

Phoneme Context Distance

- D_p is phoneme context distance
  - Distance is 0 if phonemic categories are the same (e.g. /P/ and /P/)
  - Distance is 1 if viseme classes are different (e.g. /P/ and /IY/)
  - Distance is between 0 and 1 if different phonemic classes but same viseme class (e.g. /P/ and /B/)
- Compute for the entire triphone
- Weight the center phoneme most

Lip Shape Distance

- D_s is distance between lip shapes in overlapping triphones
  - Eg. for “teapot”, contours for /IY/ and /P/ should match between /T-IY-P/ and /IY-P-AA/
  - Compute Euclidean distance between 4-element vectors (lip width, lip height, inner lip height, height of visible teeth)
- Solution depends on neighbors in both directions (use D_P)

Time Alignment of Triphone Videos

Need to combine triphone videos

Choose portion of overlapping triphones where lip shapes are close as possible

Already done when computing D_s

Time Alignment to Utterance

Still need to time align with target audio

- Compare corresponding phoneme transcripts
- Start time of center phoneme in triphone is aligned with label in target transcript
- Video is then stretched/compressed to fit time needed between target phoneme boundaries

Combining Lips and Background

Need to stitch new mouth movie into background original face sequence

Compute transform M as before

Warping replacement mask defines mouth and background portions in final video

Combining Lips and Background

- Mouth shape comes from triphone image, and is warped using M
- Jaw shape is combination of background jaw and triphone jaw lines
- Near ears, jaw dependent on background, near chin, jaw depends on mouth
- Illumination matching is used to avoid seams mouth and background

Video Rewrite Results

- Video: 8 minutes of video, 109 sentences
- Training Data: front-facing segments of video, around 1700 triphones

Video Rewrite Results

2 minutes of video, 1157 triphones



Video Rewrite

- Image-based facial animation system
- Driven by audio
- Output sequence created from real video
- Allows natural facial movements (eye blinks, head motions)

Making Faces

- Allows capturing facial expressions in 3D from a video sequence
- Provides a 3D model and texture that can be rendered on 3D hardware

Data Capture

- Actor's face digitized using a Cyberware scanner to get a base 3D mesh
- Six calibrated video cameras capture actor's expressions

Data Capture

182 dots are glued to actor's face

Each dot is one of six colors with fluorescent pigment

Dots of same color are placed as far apart as possible

Dots follow the contours of the face (eyes, lips, nasio-labial furrows, etc.)

Dot Labeling

- Each dot needs a unique label
- Dots will be used to warp the 3D mesh
- Also used later for texture generation from the six views
- For each frame in each camera:
  - Classify each pixel as belonging to one of six categories
  - Find connected components
  - Compute the centroid

Dot Labeling (continued)

Need to compute dot correspondences between camera views

Must handle occlusions, false matches

Compute all point correspondences between k cameras and n 2D dots

$$\binom{k}{2} n^2 \quad \text{point correspondences}$$

Dot Labeling (continued)

- For each correspondence
  - Triangulate a 3D point based on closest intersection of rays cast through 2D dots
  - Check if back-projection is above some threshold
  - All 3D candidates below threshold are stored

Dot Labeling (continued)

Project stored 3D points into a reference view

Keep points that are within 2 pixels of dots in reference view

These points are potential 3D matches for a given 2D dot

Compute average as final 3D position

Assign to 2D dot in reference view

Dot Labeling (continued)

- Need to assign consistent labels to 3D dot locations across entire sequence
- Define a reference set of dots *D* (frame 0)
- Let $d_j \in D$ be the neutral location for dot *j*
- Position of d_j at frame *i* is $d_j^i = d_j + v_j^i$
- For each reference dot, find the closest 3D dot of same color within some distance ε

Moving the Dots

Move reference dot to matched location

For unmatched reference dot d_k, let n_k be the set of neighbor dots with match in current frame i

$$v_k^i = \frac{1}{\|n_k\|} \sum_{d_j^i \in n_k} v_j^i$$

Constructing the Mesh

- Cyberware scan has problems:
  - Fluorescent markers cause bumps on mesh
  - No mouth opening
  - Too many polygons
- Bumps removed manually
- Split mouth polygons, add teeth and tongue polygons
- Run mesh simplification algorithm (Hoppe's algorithm: 460k to 4800 polys)

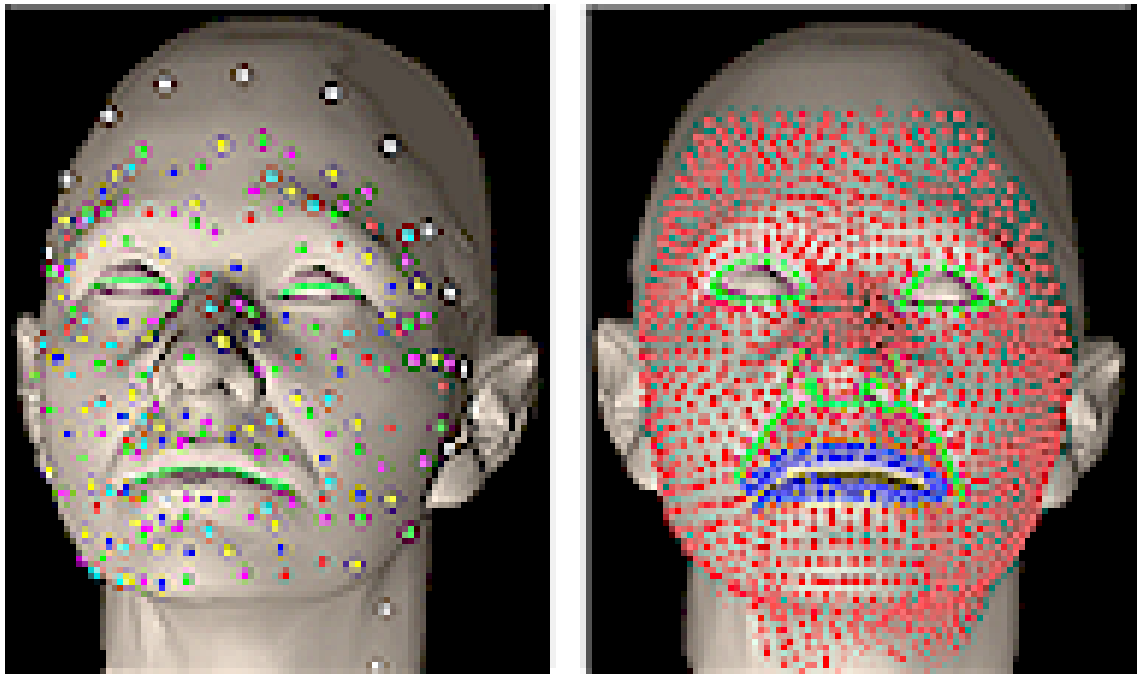
Moving the Mesh

Move vertices by linear combination of offsets of nearest dots

$$p_j^i = p_j + \sum_k \alpha_k^j \left\| d_k^i - d_k \right\|$$

Assigning Blend Coefficients

Assign blend coefficients for a grid of 1400 evenly distributed points on face



Assigning Blend Coefficients

- Label each dot, vertex, grid point as *above*, *below*, or *neither*
- Find 2 closest dots to each grid point *p*
- D_n is set of dots within $1.8(d_1+d_2)/2$ of *p*
- Remove points in relatively same direction
- Assign blend values based on distance from *p*

Assign Blend Coefficients (cont.)

If dot not in Dn, then a is 0

If dot in Dn:

$$\text{let } l_i = \frac{1.0}{\|d_i - p\|}, \text{ then } \alpha_i = \frac{l_i}{\sum_{d_i \in D_n} l_i}$$

- For vertices, find closest grid points
- Copy blend coefficients

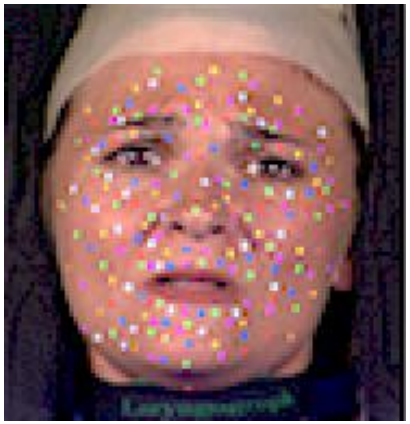
Dot Removal

Substitute skin color for dot colors

First low-pass filter the image

Directional filter prevents color bleeding

Black=symmetric, White=directional



Face with dots

Low-pass mask

Dot Removal (continued)

Extract rectangular patch of dot-free skin

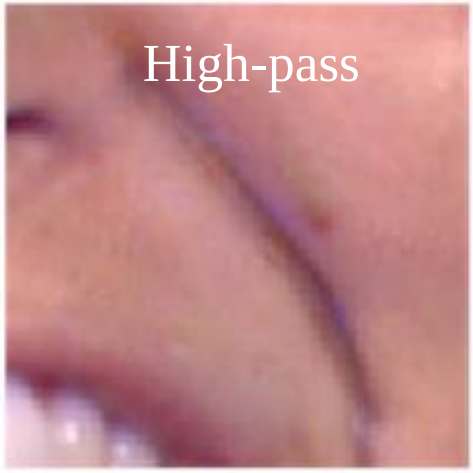
High-pass filter this patch

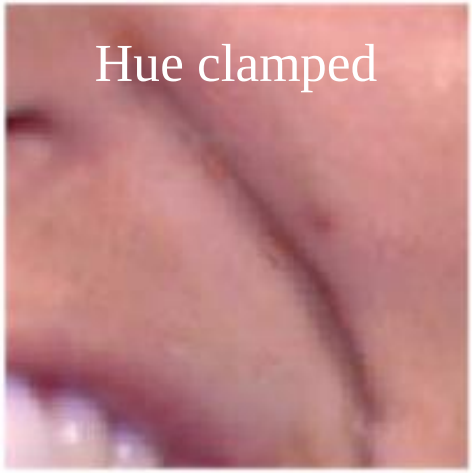
Register patch to center of dot regions

Blend it with the low-frequency skin

Clamp hue values to narrow range

Dot Removal (continued)





Texture Generation

Texture map generated for each frame

- Project mesh onto cylinder
- Compute mesh location (k, β_1, β_2) for each texel (u,v)
- For each camera, transform mesh into view
- For each texel, get 3D coordinates in mesh
- Project 3D point to camera plane
- Get color at (x,y) and store as texel color

Texture Generation (continued)

Compute a texel weight (dot product between texel normal on mesh and direction to camera)

Merge the texture maps from all the cameras based on the weight map

Results

Making Faces Summary

3D geometry and texture of a face

Data generated for each frame of video

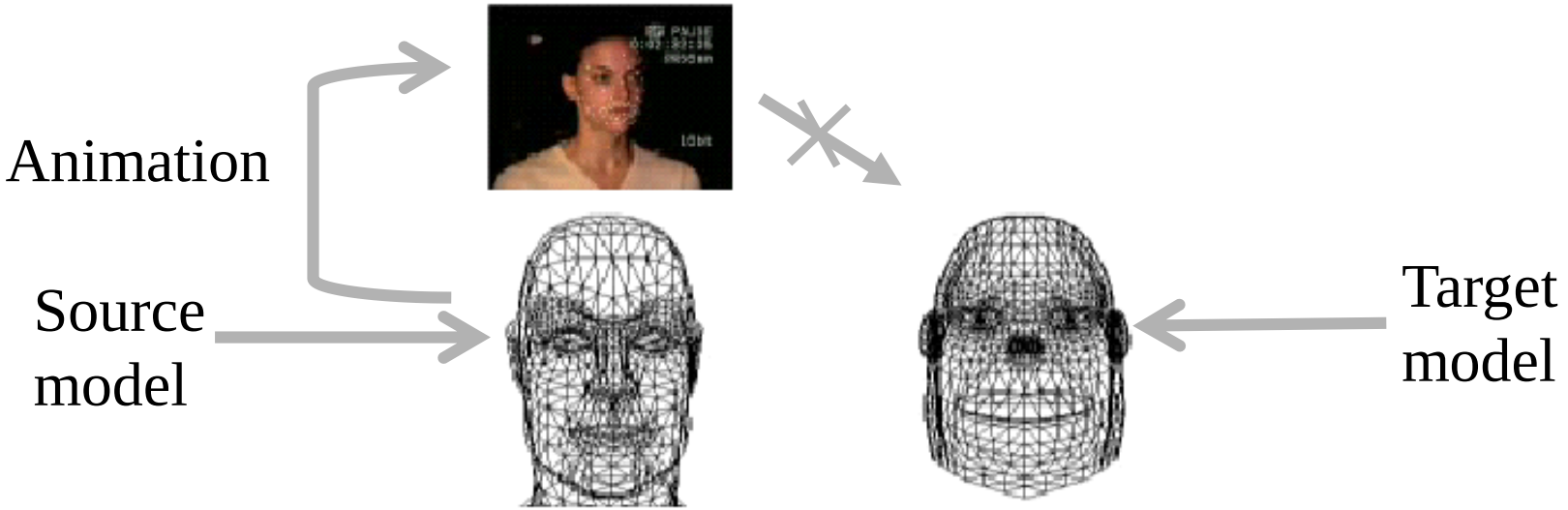
Shading and highlights “glued” to face

Nice results, but not totally automated

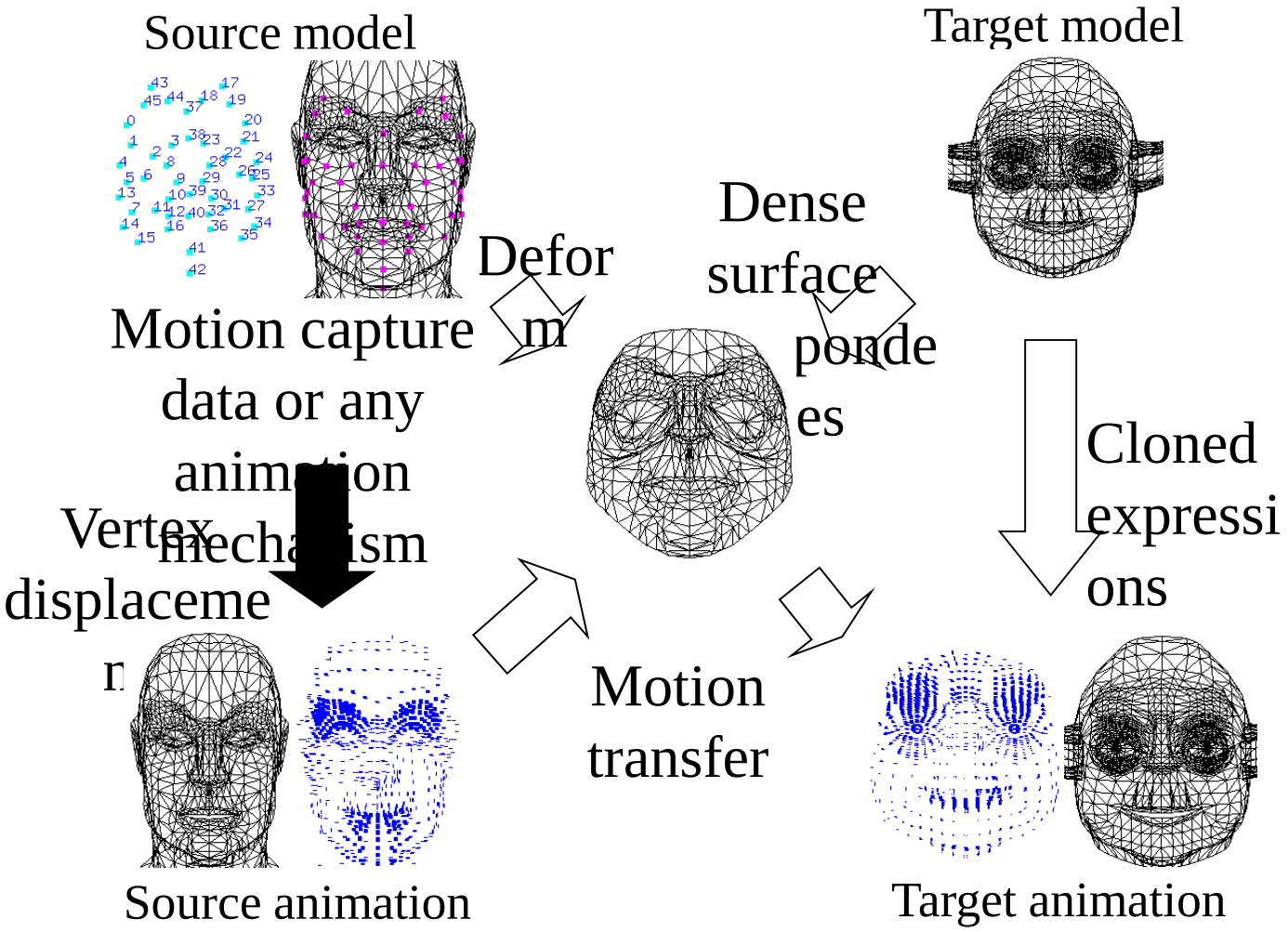
Need to repeat entire process for every new face we want to animate

Expression Cloning

Allows facial expressions to be mapped from one model to another



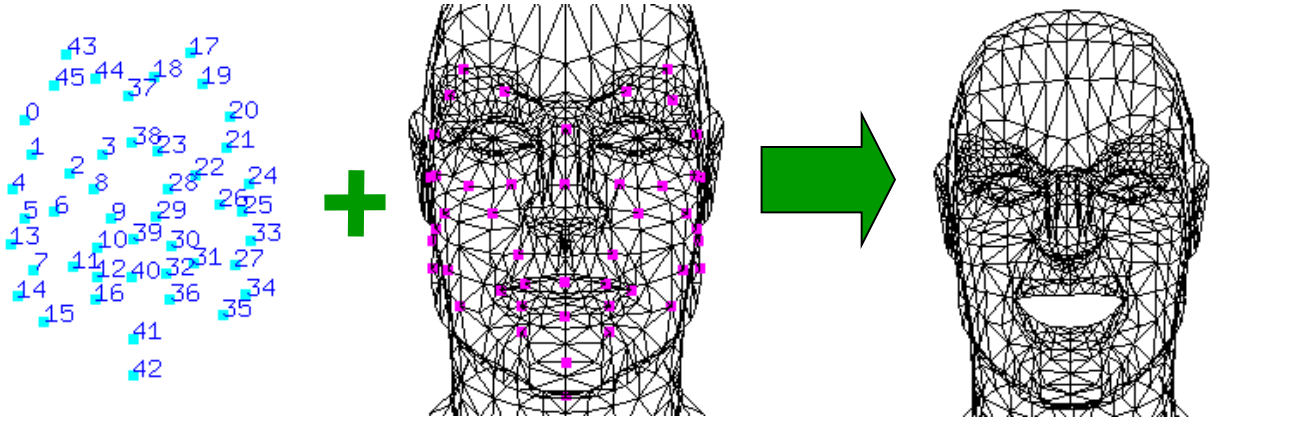
Expression Cloning Outline



Source Animation Creation

Use any existing facial animation method

E.g. “Making Faces” paper described earlier



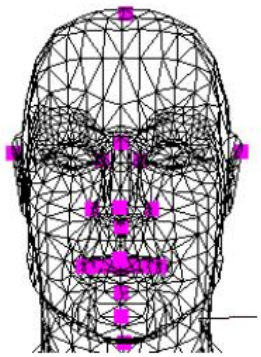
Dense Surface Correspondence

Manually select 15-35 correspondences

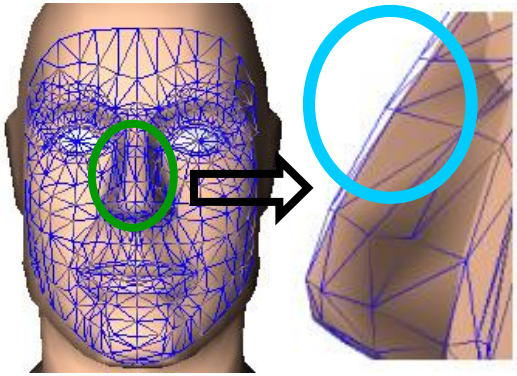
Morph the source model using RBFs

Perform cylindrical projection (ray through source vertex, into target mesh)

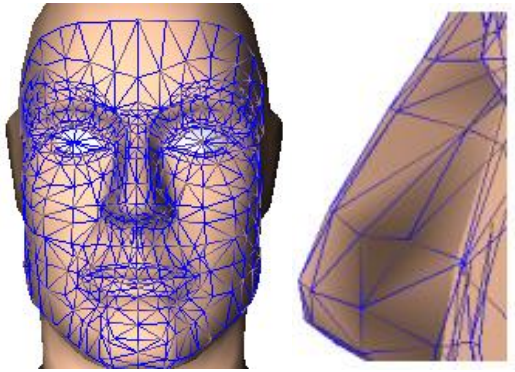
Compute Barycentric coords of intersection



Initial Features



After RBF



After projection

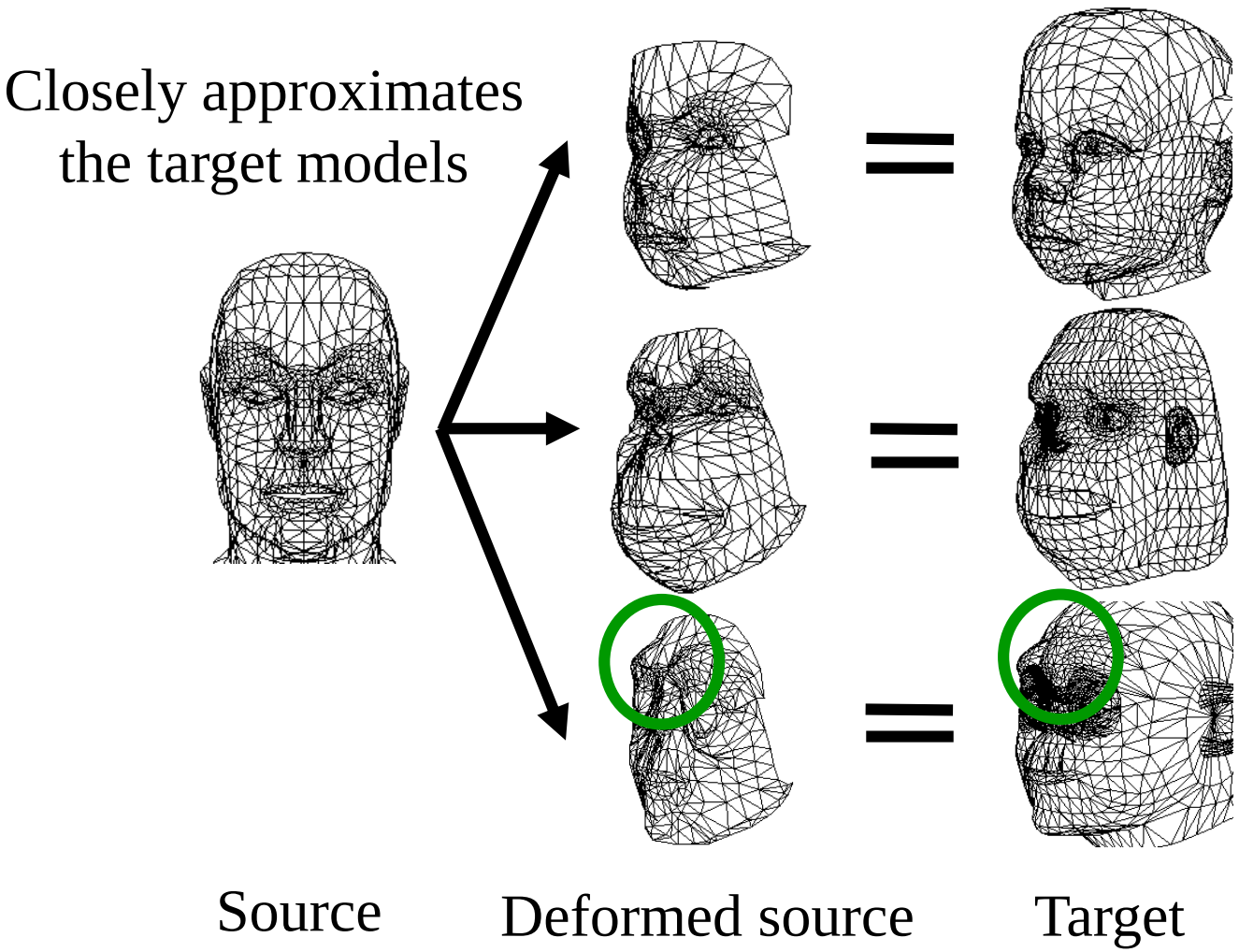
Automatic Feature Selection

Can also automate initial correspondences

Use basic facts about human geometry:

- Tip of Nose = point with highest Z value
- Top of Head = point with highest Y value
- Currently use around 15 such heuristic rules

Example Deformations

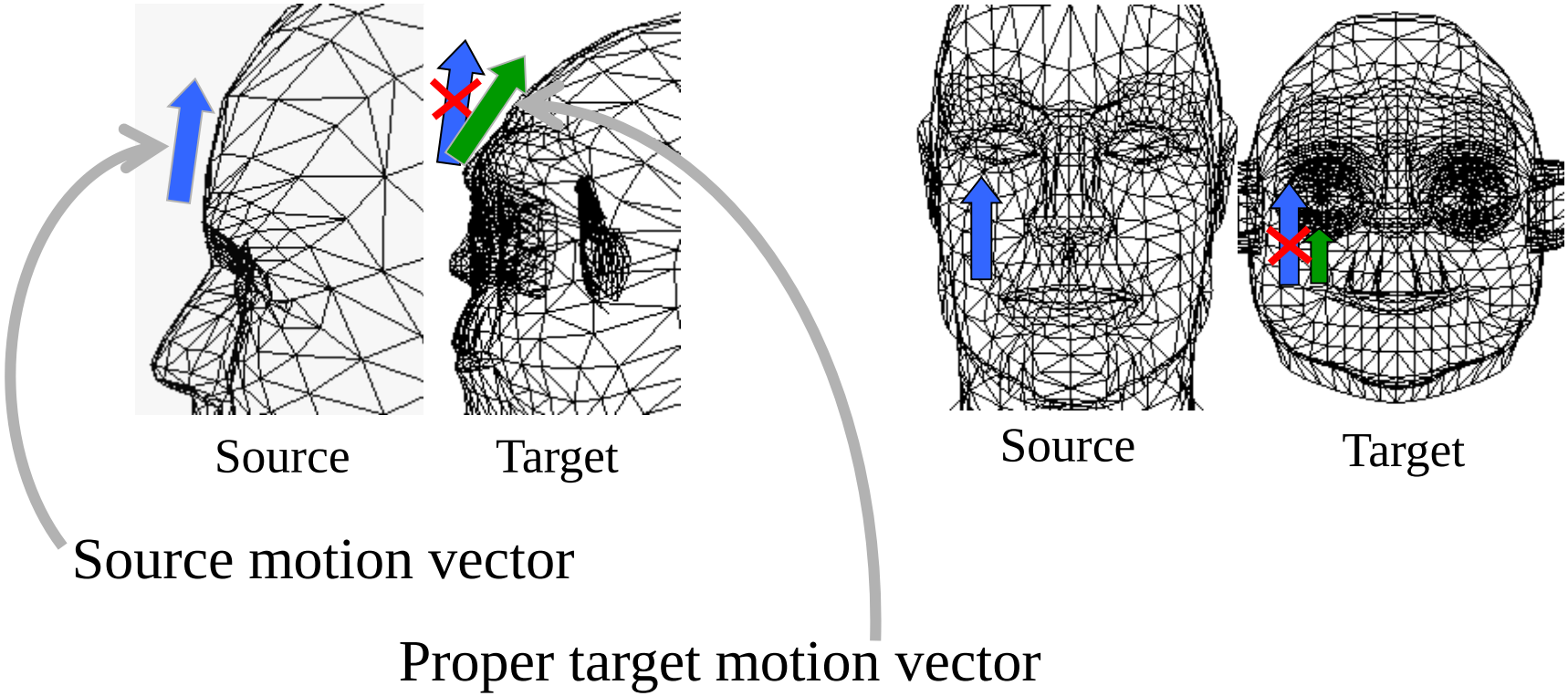


Animation with Motion Vectors

- Animate by displacing target vertex by motion of corresponding source point
- Interpolate Barycentric coordinates of target vertices based on source vertices
- Need to project target model onto source model (opposite of what we did before)

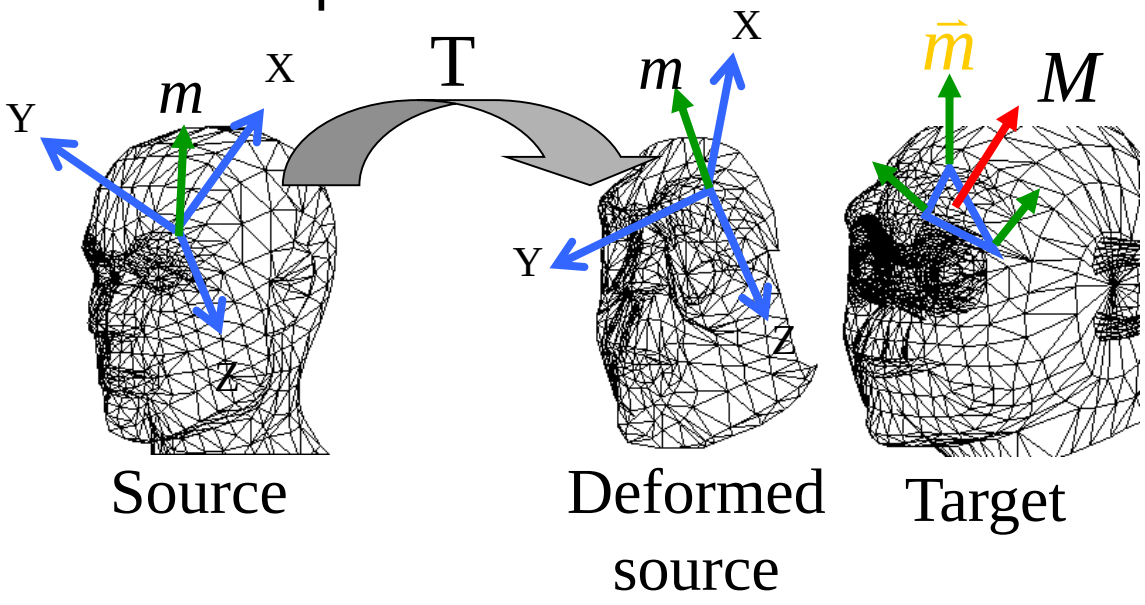
Motion Vector Transfer

Need to adjust direction and magnitude



Motion Vector Transfer (cont.)

- Attach local coordinate system for each vertex in source and deformed source
- X-axis = average normal
- Y-axis = proj. of any adjacent edge onto plane with X-axis as normal
- Z-axis = cross product of X and Y

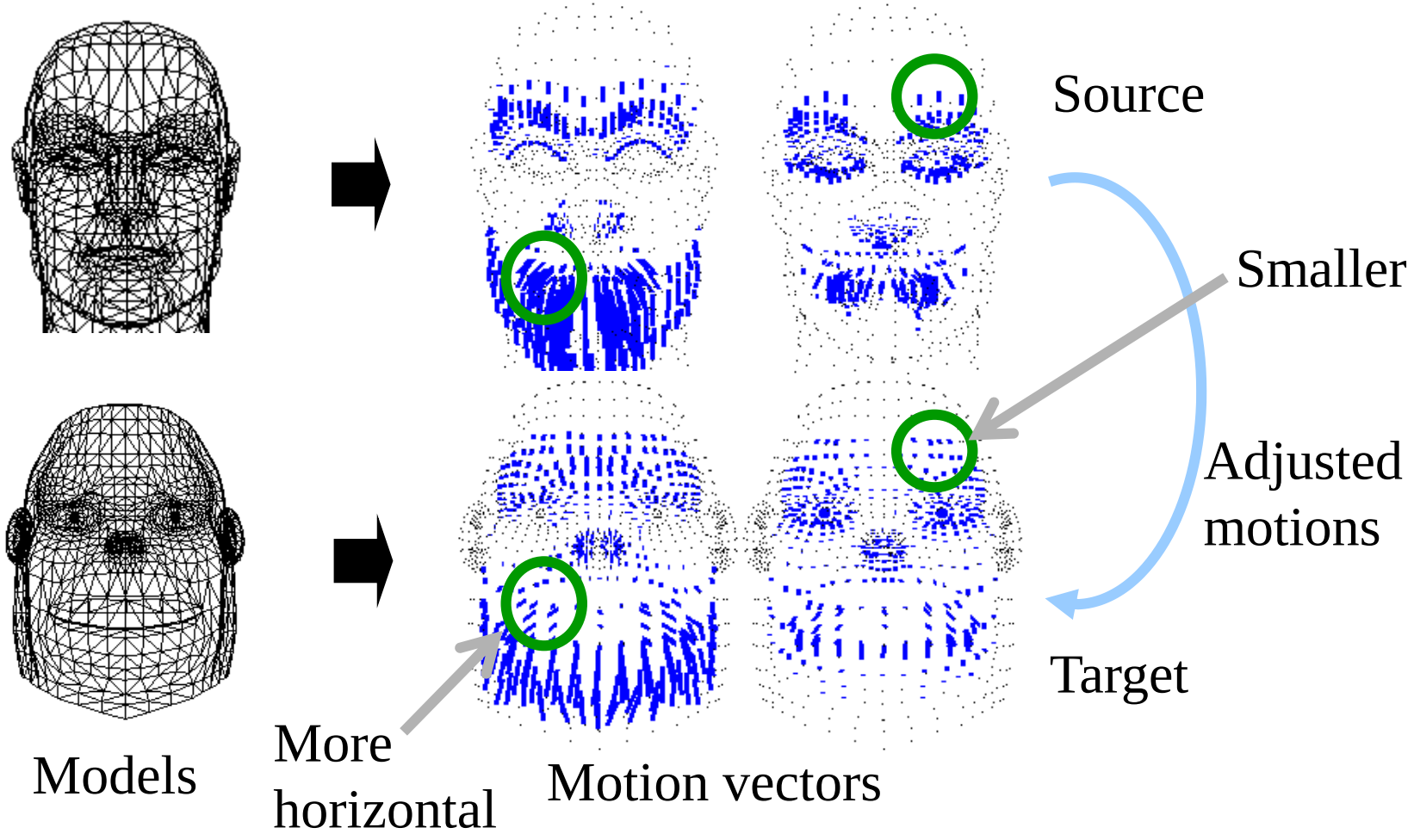


Motion Vector Transfer

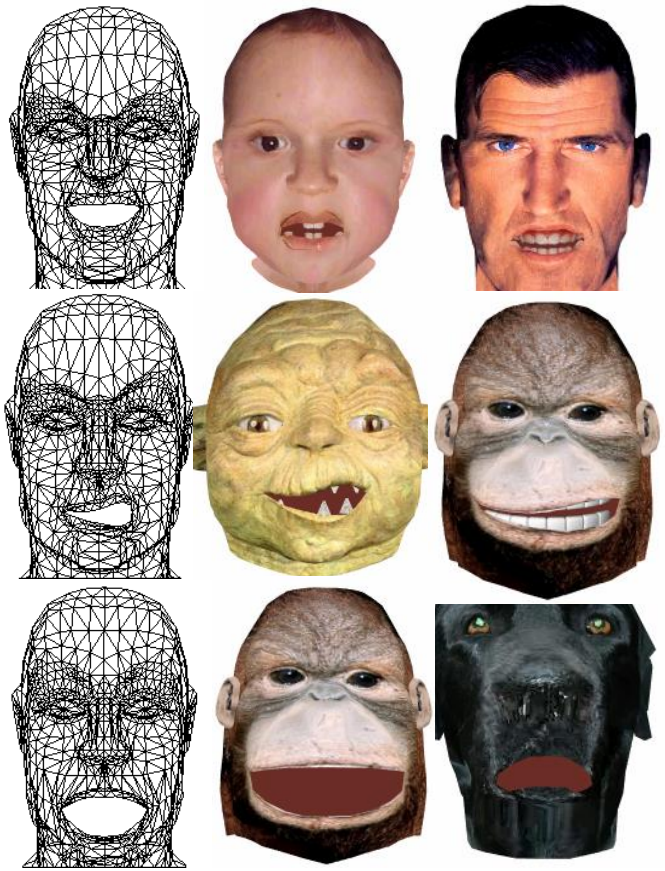
Compute transformation between two coordinate systems

Mapping determines the deformed source model motion vectors

Example Motion Transfer



Results



Angry expression

Distorted mouth

Big open mouth

Source Targets

Summary

- EC can animate new models using a library of existing expressions
- Transfers motion vectors from source animations to target models
- Process is fast and can be fully automated